

**REMARKS**

Claims 61-79 are pending in the present application. Reconsideration of the claims is respectfully requested.

**I. 35 U.S.C. §102, anticipation**

The Examiner has rejected claims 61-67, 69-75 and 77-79 under 35 USC §102(e) as being anticipated by Schaefer (US 2003/0084429). This rejection is respectfully traversed.

A prior art reference anticipates the claimed invention under 35 U.S.C. § 102 only if every element of a claimed invention is identically shown in that single reference, arranged as they are in the claims. *In re Bond*, 910 F.2d 831, 832, 15 U.S.P.Q.2d 1566, 1567 (Fed. Cir. 1990). All limitations of the claimed invention must be considered when determining patentability. *In re Lowry*, 32 F.3d 1579, 1582, 32 U.S.P.Q.2d 1031, 1034 (Fed. Cir. 1994).

Claim 61 recites:

61. A method for importing specifications from a product design document for use in manufacturing testing software, the method comprising the computer implemented steps of:
- (a) extracting test specification data from an electronic design document, wherein the design document is in a structured format containing both data and metadata;
  - (b) translating the extracted test specification data into a format that is readable by the testing software; and
  - (c) importing the translated test specification data into test executive software that manages testing and passes test specifications into applicable sections of underlying test library software that performs specific test functions.

Claims 69 and 77 contain similar limitations for purposes of the present discussion.

The Schaefer reference cited by the Examiner appears to have some superficial similarities with the present invention in as much as it is directed toward product testing software. However, a closer reading of Schaefer reveals that it does not in fact teach all of the limitations of the claimed invention.

First, the claimed invention is directed toward product testing within a manufacturing environment. In contrast, Schaefer is directed toward software development, not manufacturing. The process of writing software does not constitute manufacturing according to any common or reasonable use of that term.

Furthermore, the process taught in Schaefer does involve an electronic product design document from which test specifications are drawn. Schaefer uses GUI maps, which are files that contain information about user interface objects. These GUI maps are then translated into database tables. The GUI maps relate to the organization of the software and are used to locate software objects. The GUI maps do not contain any normative data regarding product testing specifications and are not equivalent to the design document of the claimed invention.

A separate data input component is used by Schaefer to create test cases to be entered into the tables created from the GUI maps. The tables allow the test cases to be matched with the correct execution paths. Schaefer does teach importing data from a structured document like a spread sheet (paragraph [0053]). However, this imported data is merely example input data used to run specific execution paths and does not constitute normative testing specifications taken from a product design document.

Because claims 62-67, 68-75, and 78-79 depend from claims 61, 69, and 77 respectively, they are distinguished from Schaefer for the reasons explained above. Furthermore, the dependent claims include limitations not taught or suggested by Schaefer. For example, claims 62, 63, 66 and 67 recite:

62. The method according to claim 61, wherein the test specifications include at least one of the following:
- product platform;
  - product code;
  - revision number;
  - product characteristics;
  - list of tests including test sequence;
  - test name;
  - test method;
  - test description;
  - test conditions;
  - test limits; and
  - resolution of digits.

63. The method according to claim 61, wherein step (a) further comprises assembling a variable list containing test parameter values from said test specifications.

66. The method according to claim 61, wherein the test specifications imported into the test executive include test specification types that are pre-defined by the test executive software.

67. The method according to claim 61, wherein the test specifications imported into the test executive include user-defined test specification types.

These claims relate to normative testing data for product specifications which is not taught by Schaefer, as explained above.

Therefore, Applicant respectfully asserts that the rejection of claims 61-67, 69-75 and 77-79 under 35 U.S.C. § 102 has been overcome.

## **II. 35 U.S.C. §103, obviousness**

The Examiner has rejected claims 68 and 76 under 35 USC §102(e) as being anticipated by Schaefer (US 2003/0084429). This rejection is respectfully traversed.

A prima facie case of obviousness is established when the teachings of the prior art itself suggest the claimed subject matter to a person of ordinary skill in the art. *In re Bell*, 991 F.2d 781, 783, 26 U.S.P.Q.2d 1529, 1531 (Fed. Cir. 1993). In a proper obviousness determination, regardless of whether the changes from the prior art are "minor," the changes must be evaluated in terms of the whole invention, including whether the prior art provides any teaching or suggestion to one of ordinary skill in the art to make the changes that would produce the claimed invention. *In re Chu*, 66 F.3d 292, 298, 36 U.S.P.Q.2d 1089, 1094 (Fed. Cir. 1995).

Because claims 68 and 76 depend from claims 61 and 69, respectively, they are distinguished from Schaefer for the reasons stated above. Furthermore, there is no teaching in Schaefer that suggests the limitations of claims 68 and 76, which recite:

68. The method according to claim 61, further comprising:

(d) debugging and editing the test specifications with the test executive software in the event said design document contains mistakes; and

(e) exporting the edited test specifications into a second, revised design document.

76. The computer program product according to claim 69, further comprising:

(d) fourth instructions for debugging and editing the test specifications with the test executive software in the event said design document contains mistakes; and

(e) fifth instructions for exporting the edited test specifications to a second revised design document.

As explained above, Schaefer does not teach or suggest using a product design document and associated normative specifications.

Therefore, Applicant respectfully asserts that the rejection of claims 68 and 76 under 35 U.S.C. § 103 has been overcome.

### III. Response to Arguments

In response to the above arguments, the Examiner has asserted:

In regard to independent claim 61, the Applicant argues that the claimed invention is directed toward product testing within a manufacturing environment while the Schaefer reference was directed toward software testing in a software development environment. In response to applicant's argument that claimed invention is specifically directed toward product testing within a manufacturing environment, the Examiner notes that a recitation of the intended use of the claimed invention must result in a structural difference between the claimed invention and the prior art in order to patentably distinguish the claimed invention from the prior art. If the prior art structure is capable of performing the intended use, then it meets the claim.

Additionally, in response to applicant's arguments, the recitation, "manufacturing testing software" or "product design document") [sic] (emphasis added), has not been given patentable weight because the recitation occurs in the preamble. A preamble is generally not accorded any patentable weight where it merely recites the purpose of a process or the intended use of a structure, and where the body of the claim does not depend on the preamble for completeness but, instead, the process steps or structural limitations are able to stand alone. See *In re Hirao*, 535 F.2d 67, 190 USPQ 15 (CCPA 1976) and *Kropa v. Robie*, 187 F.2d 150, 152, 88 USPQ 478, 481 (CCPA 1951).

The reference to manufacturing testing software in the preamble of the independent claims does more than simply recite the intended purpose of the process described by the claim limitations. It also specifies the technical field in which those limitations are applied. The present invention applies to software used to test the manufacture of physical products. Schaefer teaches using a system for testing the development of software products. These different technical fields of necessarily involve functional and structural differences between the claimed invention and Schaefer. A process used to test the development of software products cannot be trivially applied to testing the manufacture of physical products.

The Examiner goes on to assert:

Applicant also argues that the Schaefer reference does not involve an electronic product design document from which the test specifications are drawn. The Examiner respectfully disagrees with the Applicant. Schaefer clearly teaches wherein data import utility may import test case data from a Microsoft Excel spreadsheet (Page 4: Paragraph 53: "Alternatively, user ... Microsoft Excel file"; Page 9: Paragraph 105: "In addition ... in the database"). The data was then imported into one of the tables created by the GUI maps, wherein the data in the tables may include one or more test cases for the software program (Page 3: Paragraph 50: "The data in tables ... during execution).

Test case data referred to in Schaefer is not the same as the test specification data used in the present invention. A close examination of the paragraphs cited by the Examiner reveals this distinction:

[0050] The data in tables 240 may include one or more test cases, which define sequences or paths of transitions that the software program takes during execution. The tables 240 may include a PAGE\_ABBR table and a PAGE\_FLOW table. PAGE\_ABBR table may store an abbreviated name and a logical name for each window of the software program 185. Each row in the PAGE\_FLOW table may represent an execution path for testing a software program 185 and may correspond to a particular test case. The PAGE\_ABBR table may include a column PAGE\_ABBR with data that corresponds to data in a Page Sequence section of the PAGE\_FLOW table.

[0051] In addition, a GUI translator component may create three tables for each GUI map associated with a software program 185. The combination of the data in the three tables may specify one or more sub-paths for the execution of a window in the software program 185. The three tables may include the following: a test data table, which may include data that may

be input into objects on the window associated with the GUI map during execution of the software program 185; a seized data table, which may include data that may instruct the test engine component to verify whether data selected on one window is accurately displayed on another window; and an object data table, which may include columns that match the logical names of Action" data objects that cause the software program 185 to transition from an active window to a next window during execution of the software program 185.

[0052] The three tables may each include a column named FLOW\_ID, which may correspond to the column FLOW\_ID in the PAGE\_FLOW table. In addition, each of the three tables may include a column PAGE\_ID, which corresponds to the data in the Page Sequence section of the PAGE\_FLOW table.

[0053] User 210 may access a data input component 165 to input data 260 into one or more of the tables 240. The data 260 may include information that may be used by test engine component 170 to test the software program 185. Alternatively, user 210 may access a data import utility to import data into one or more of the tables 240. Data import utility may import data 260 from, for example, a Microsoft Excel file.

[0054] User 210 may access the test engine component 170 to request testing of the software program 185. Test engine component 170 may retrieve data from one or more of the tables 240, wherein the data specifies the execution paths for testing the software program 185. In addition, test engine component 170 may open one or more of the generated GUI maps associated with the software program 185, and may read the contents of the GUI map into memory 120. Test engine component 170 may then call a software controller component 173 function to transmit one or more instructions and data to the software program 185 for controlling the execution of the software program 185. A software controller component 173 may include a library of functions that may be called to provide instructions and data to the software program 185. The software controller component 173 may shield the calling program, such as test engine 170 from the implementation and execution details of the software program 185. In addition, software controller component 173 may transmit results of the processing of software program 185 to the calling program. The software controller component 173 may include a commercial software controller, such as TSL provided by the Mercury Interactive Corporation.

[0055] Test engine component 170 may also use the contents of the GUI map 230 stored in memory 120 to determine which software controller component 173 function to call to process the objects on the window that may be described by the GUI map 230. Test engine component 170 may also monitor results, received from the software controller 173 about the

execution of the software program 185, and may store such results in one or more test result files 270. Optionally, test engine component 170 displays the results to user 210 or uses the results to generate one or more performance reports.

[0105] Automation system 100 may provide a data import utility, which provides the ability for a user 210 to set up a test case that is to be run repeatedly while varying the test data. In many cases the test case data for this type of testing comes from external production systems. The data input utility translates single production data fields (e.g., Housing=O, R, or X) to multiple radio buttons (Own, Rent, and Other) within the targeted application window. In addition, the data input utility may import test data into database 135. For example, the data input utility may import test case data from a Microsoft Excel spreadsheet into a source table in the database 135. Another feature of the data input utility is its ability to replicate a test case as described above while importing data into database 135.

The test case data described in Schaefer relates to software execution paths in software. Input data is provided to test these execution paths. This has absolutely nothing to do with manufacturing specifications. As is clearly described in the above sections of Schaefer, the test case data is used to track what happens as the tested software executes, and then it provides the results. None of this relates to test specifications which provide normative parameters for manufacturing.

In addition, the description of the function of the test case data further highlights the structural differences between manufacturing and software development, adding weight to the relevance of the preamble of the claims.

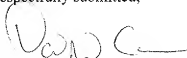
**Conclusion**

It is respectfully urged that the subject application is now in condition for allowance.

The examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

DATE: November 8, 2006

Respectfully submitted,

A handwritten signature in dark ink, appearing to read 'David W. Carstens', is written over a horizontal line.

David W. Carstens  
Reg. No. 34,134  
Carstens & Cahoon, LLP  
PO Box 802334  
Dallas, TX 75380  
(972) 367-2001  
Attorney for Applicants